

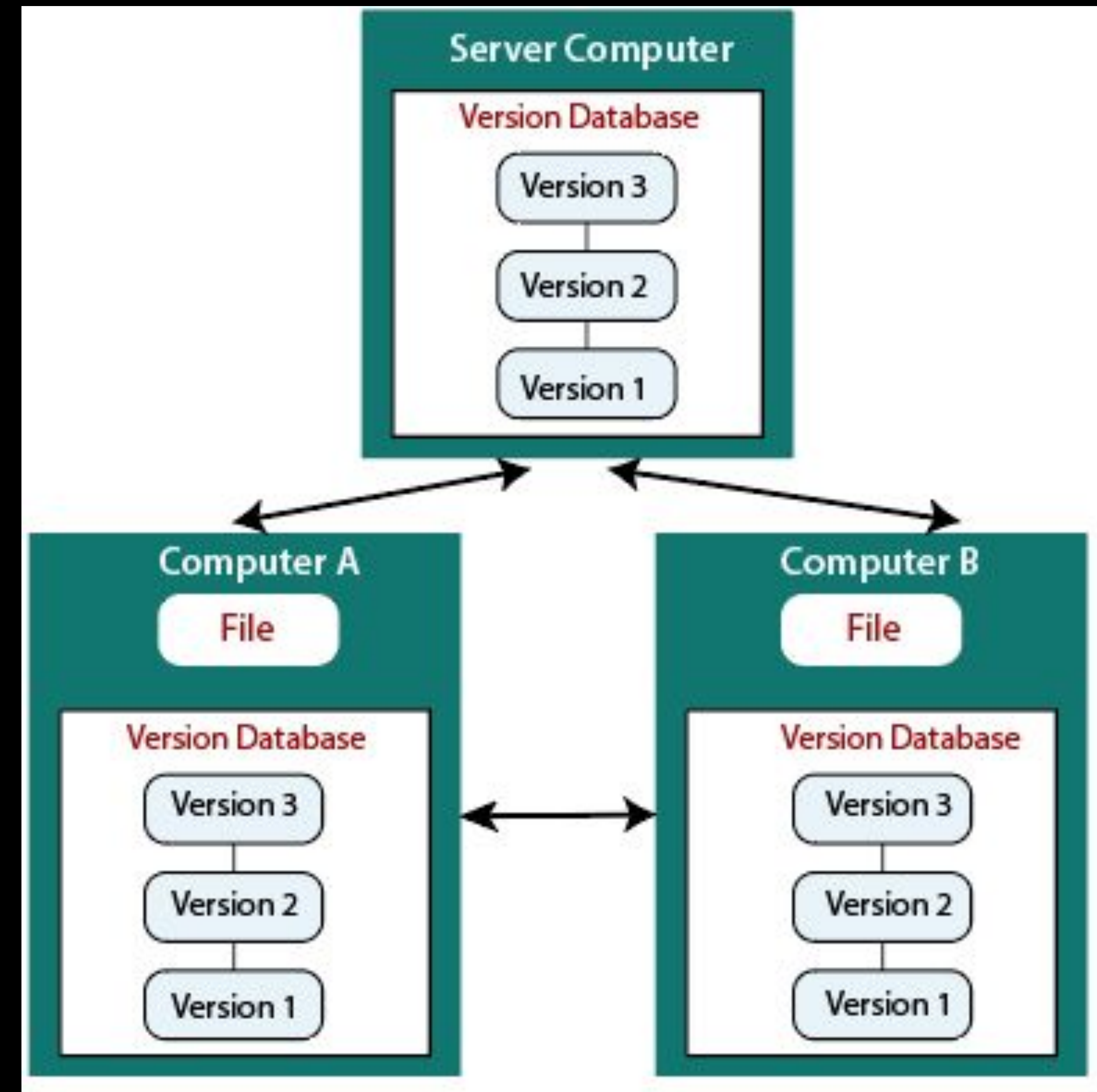
Bioinformatics of Sequence Variation

Feifei Xia Jiahui Yu

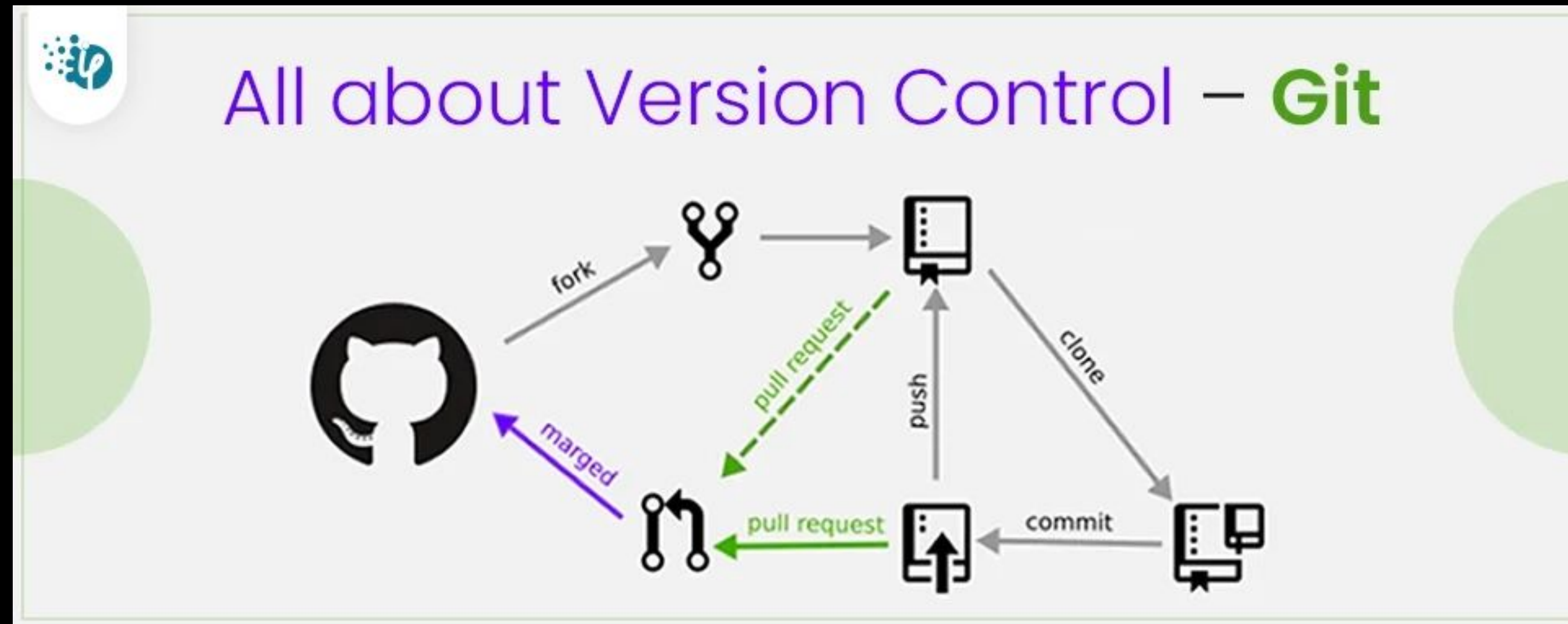
Introduction to Github

Overview of Version Control

- Version Control System (VCS) are tools that help manage changes to source code over time.
- Purpose:
 - Maintain a history of all changes
 - Facilitate collaboration by allowing multiple contributors to work on the same project without conflict.
 - Improve project organization and accountability.



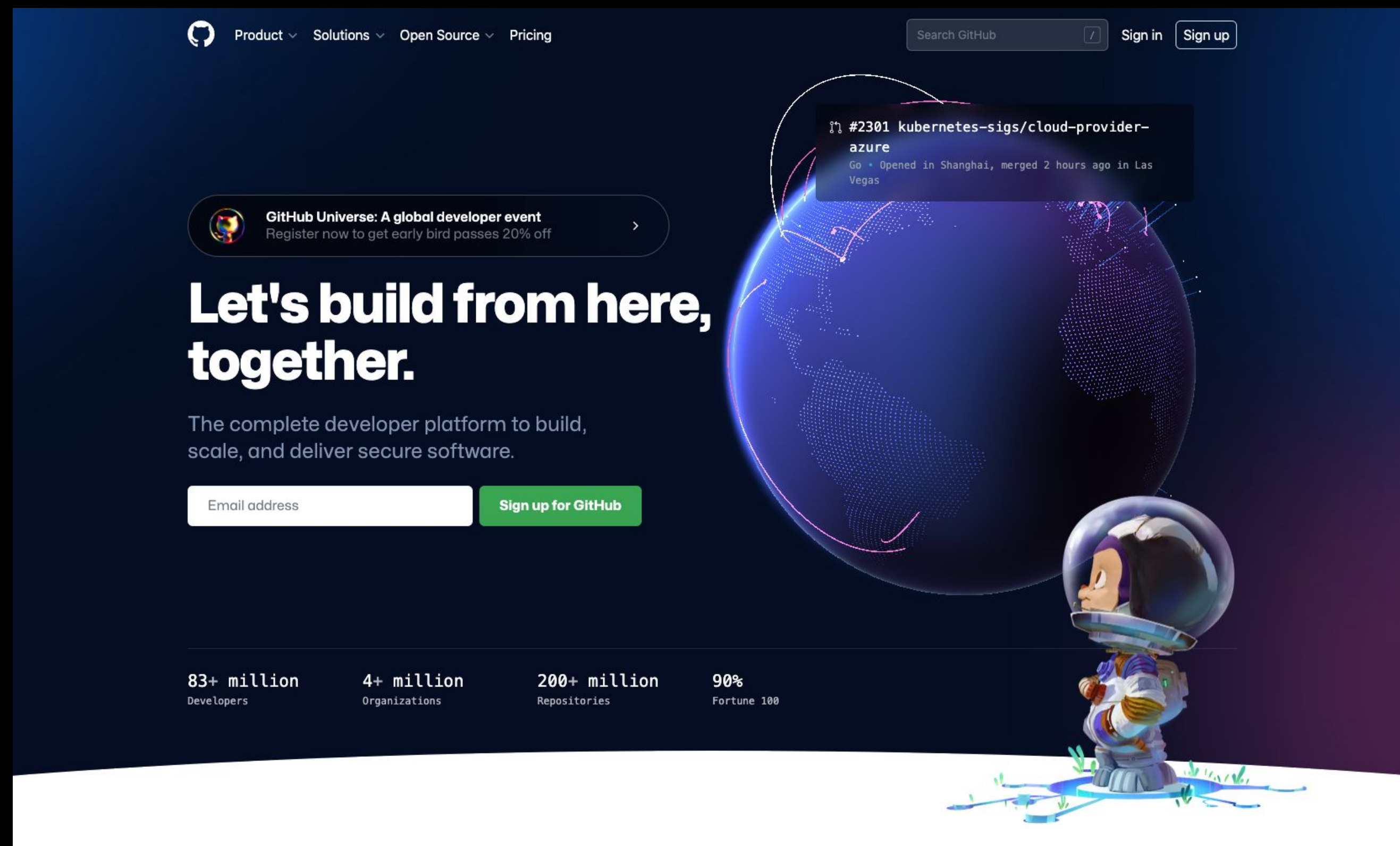
What is Git?



- Git is a distributed version control system designed to handle projects of all sizes with speed and efficiency.
- Key features:
 - Distributed: Every contributor has a complete copy of the repository, including its history, on their local machine. This enables offline work and faster operations.
 - Branching and Merging: Git allows users to create separate branches to work on features or fixes independently, then merge them back into the main codebase.

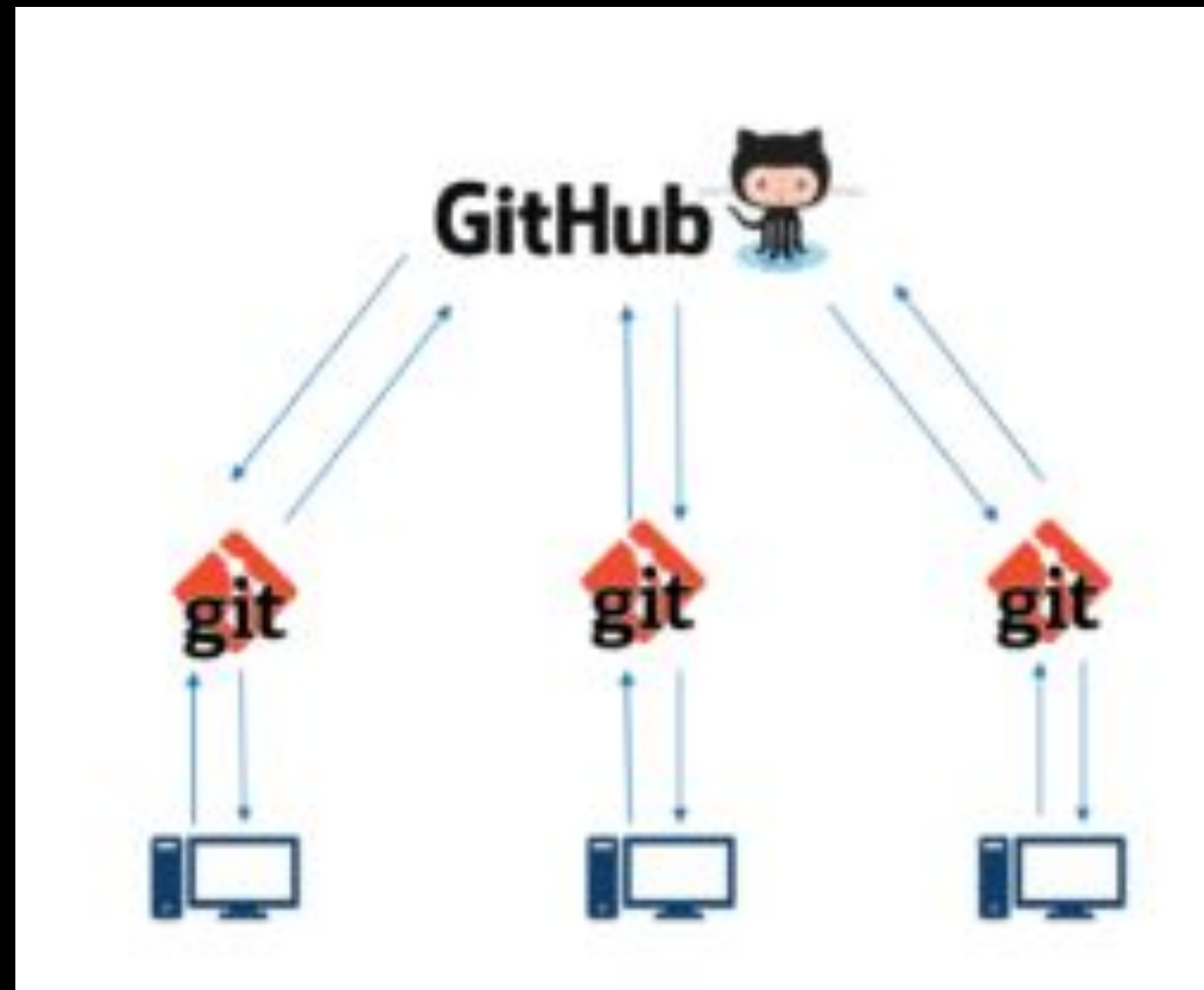
What is Github?

<https://github.com/>



- GitHub is a web-based platform that hosts Git repositories.
- To be very crisp about what exactly is GitHub, it is a file or code-sharing service to collaborate with different people.

Key Features of Github



- Remote Repositories: provides a **central place to store and manage Git repositories**, making them accessible to collaborators from anywhere.
- Collaboration Tools: features such as pull requests, code reviews, and issue tracking help teams coordinate work and maintain code quality.
- Community Engagement: Github is also a community hub for open-source projects where developers can share, contribute, and learn from each other.

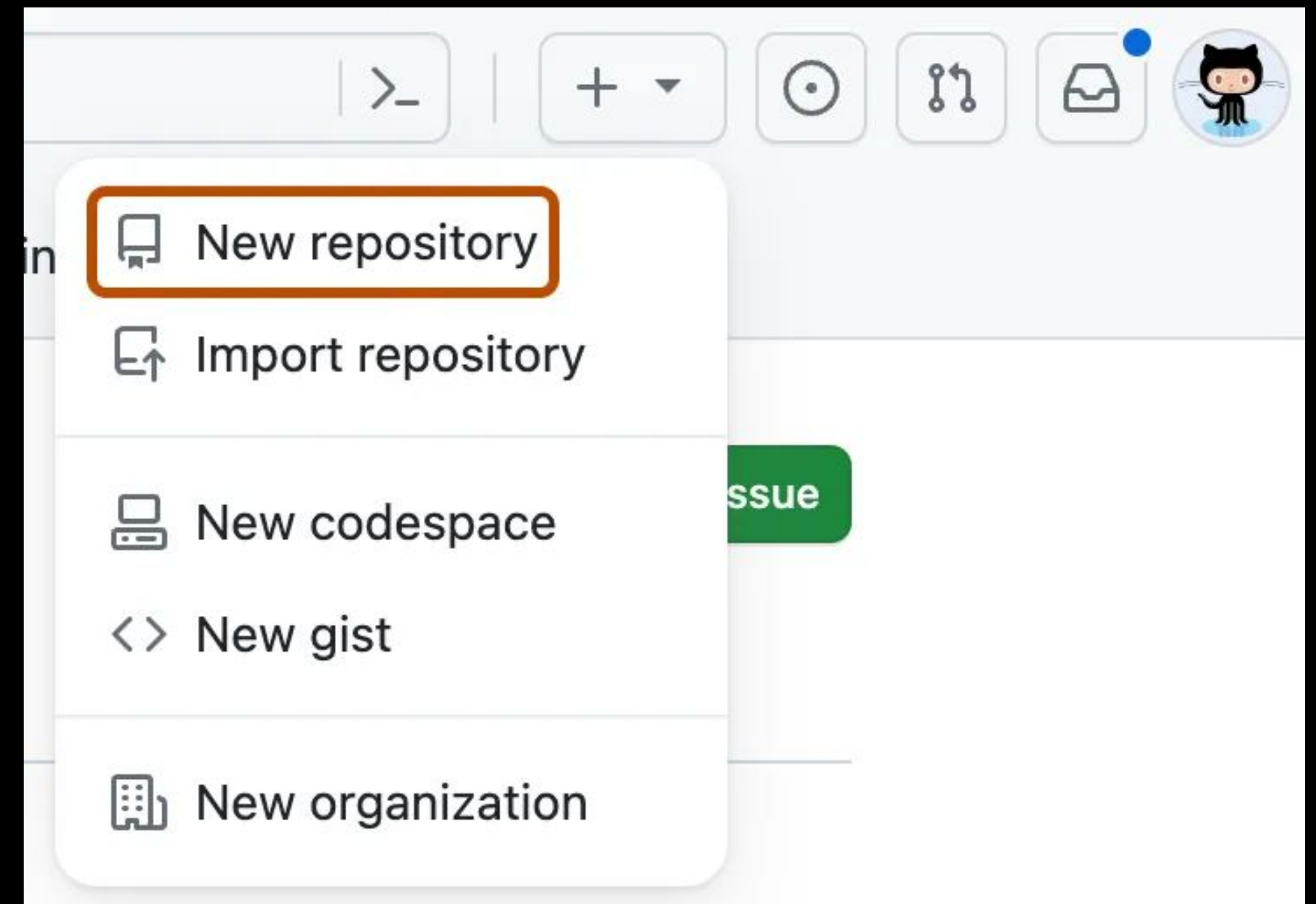
Setting up your environment

1. Create a GitHub account

- Signing Up:
 - Visit GitHub's website (<https://github.com/>) and click on “Sign up.”
 - Provide a valid email address, choose a username, and create a secure password.
- Initial Profile Setup: <https://docs.github.com/en/get-started/start-your-journey/setting-up-your-profile>
 - Verify your email to activate the account.
 - Fill out basic profile details such as your name, bio, and profile picture.
- Security Best Practices:
 - Enable two-factor authentication (2FA) to add an extra layer of security.
 - Familiarize yourself with GitHub’s privacy settings and adjust them based on your preferences.

2. Create a GitHub Repository

- In the upper-right corner of any page, select +, then click **New repository**
- In the "Repository name" box, type hello-world.
- In the "Description" box, type a short description.
For example, type "This repository is for practicing the GitHub Flow."
- Select whether your repository will be Public or Private.
- Select Add a README file.
- Click Create repository.



2. Create a GitHub Repository

3 Optionally, add a description of your repository. For example, "My first repository on GitHub."

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner

Repository name

octocat

/

hello-world

Great repository names are short and memorable. Need inspiration? How about **potential-eureka**.

Description (optional)

My first repository on GitHub

4 Choose a repository visibility. For more information, see ["About repositories."](#)

Description (optional)

☒

Public

Anyone can see this repository. You choose who can commit.

☐

Internal

Octo Corp [enterprise members](#) can see this repository. You choose who can commit.

☐

Private

You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

5 Select Initialize this repository with a README.

☒

Public

Anyone on the Internet can see this repository. You choose who can commit.

☐

Private

You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☒ Initialize this repository with a README

This will let you immediately clone the repository to your computer.

Add .gitignore: None

Add a license: None

Create repository

6 Click Create repository.

This will let you immediately clone the repository to your computer.

Add .gitignore: None

Add a license: None

Create repository

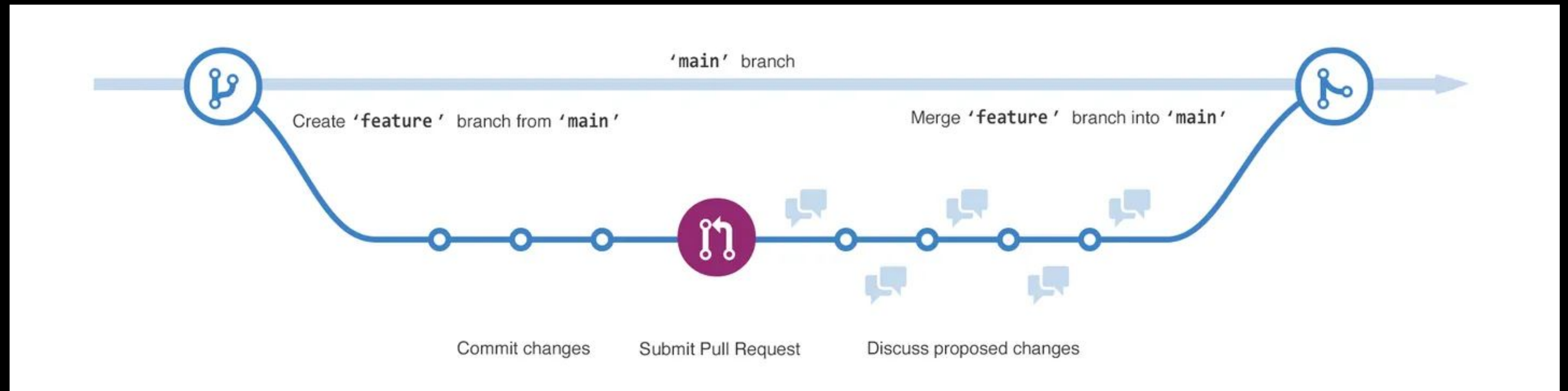
Exercise

Forward your Github accounts to bio392@compbiozurich.org

Try to create your own repository!

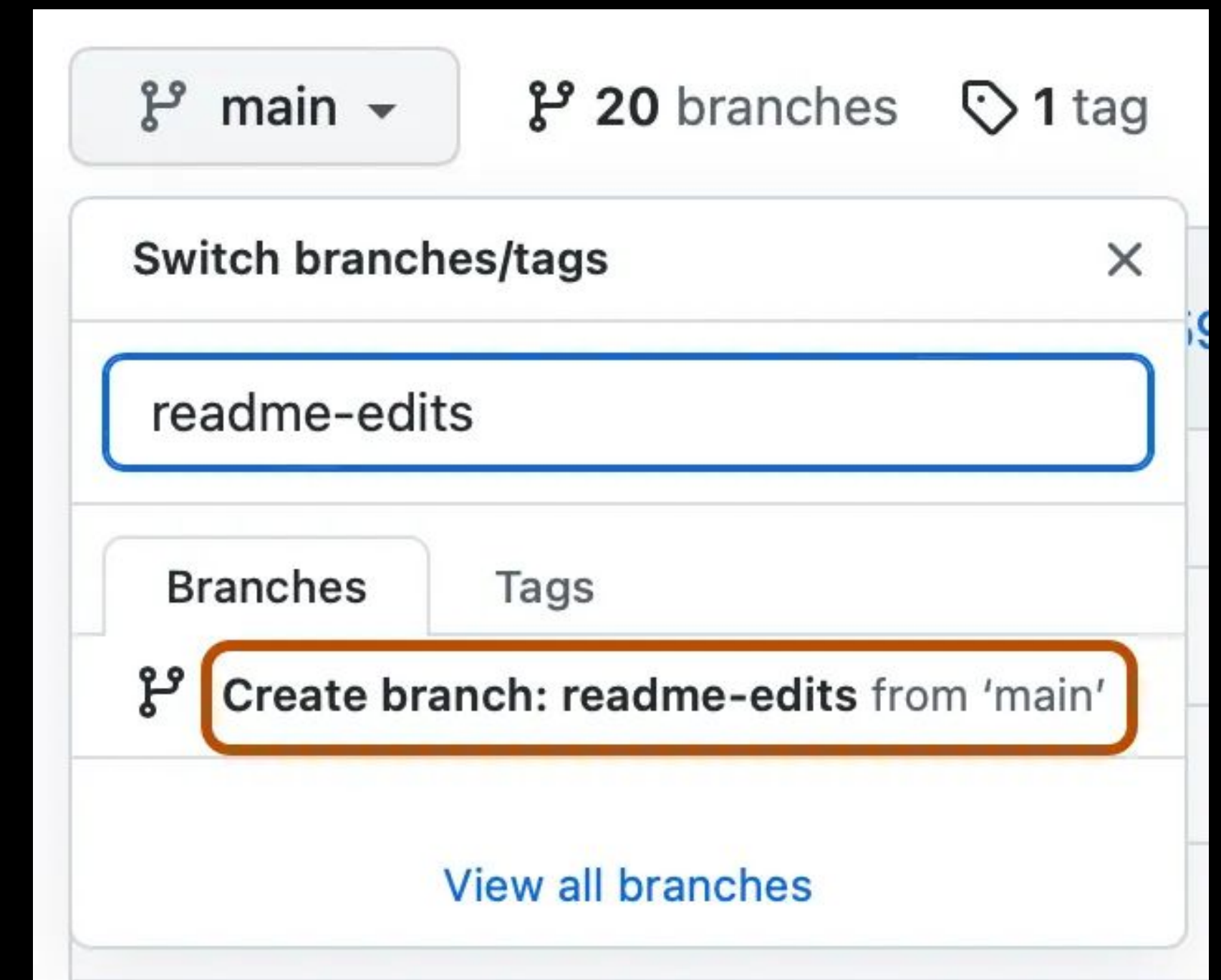
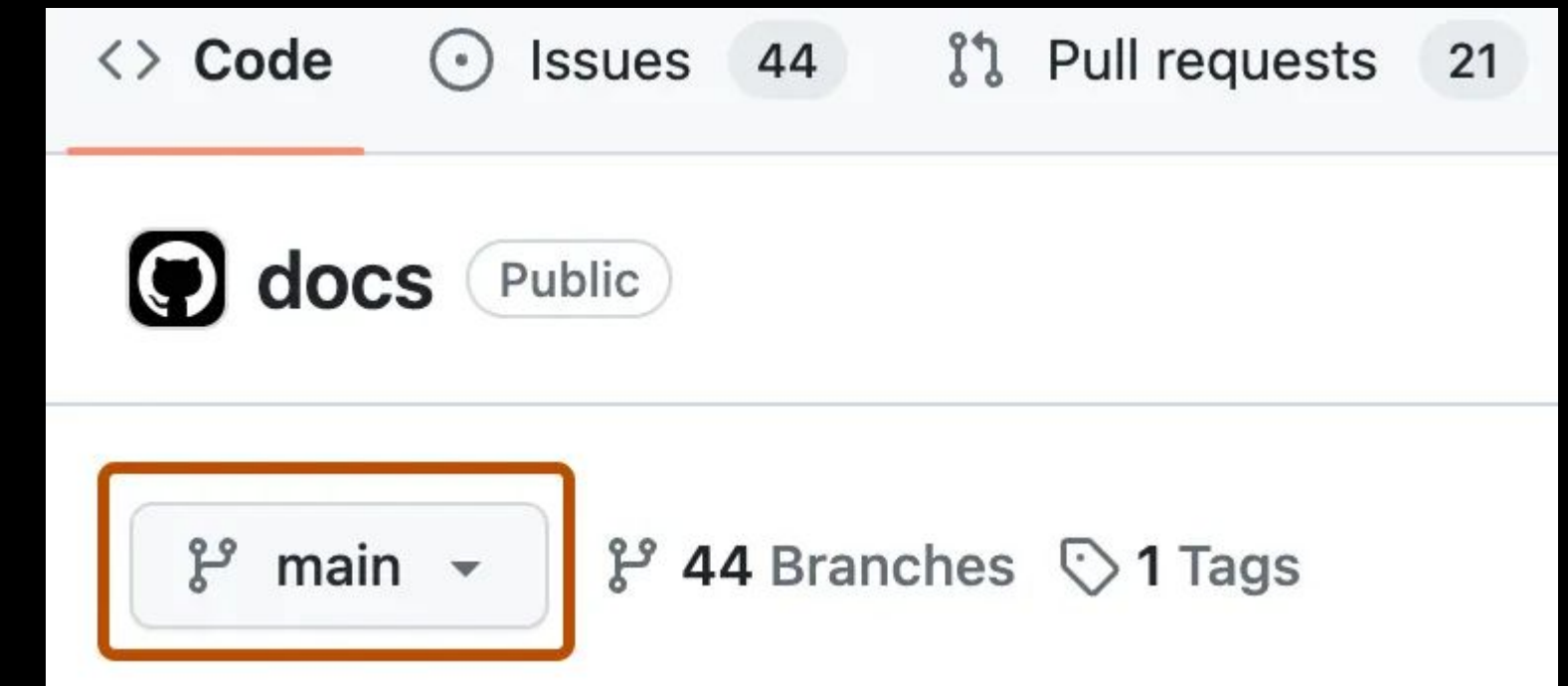
3. Create a Branch

- Branching lets you have different versions of a repository at one time.



3. Creating a branch

- Click the **Code** tab of your hello-world repository
- Above the file list, click the dropdown menu that says **main**.
- Type a branch name, **readme-edits**, into the text box
- Click **Create branch: readme-edits from main**
- Now you have two branch, main and readme-edits. When you created a new branch in the previous step, GitHub brought you to the code page for your new readme-edits branch, which is a copy of main.




Exercise

Try to create a branch!

4. Make and commit changes

- On Github, saved changes are called **commits**.
- Each commit has an associated commit message, which is a description explaining why a particular change was made.
- Commit messages capture the history of your changes so that other contributors can understand what you've done and why.

4. Make and commit changes

- 1 Under the `readme-edits` branch you created, click the `README.md` file.
- 2 To edit the file, click .
- 3 In the editor, write a bit about yourself.
- 4 Click **Commit changes**.
- 5 In the "Commit changes" box, write a commit message that describes your changes.
- 6 Click **Commit changes**.

These changes will be made only to the README file on your `readme-edits` branch, so now this branch contains content that's different from `main`.

5. Open a pull request



The screenshot shows a diff viewer interface. At the top, it says "Showing 1 changed file with 3 additions and 3 deletions." with buttons for "Split" and "Unified". Below this, the file "README.md" is shown with a line number indicator "6" and a copy icon. The diff content is as follows:

Line	Change Type	Content
...	...	@@ -1,3 +1,3 @@
1	-	# test-area-2
2	-	edit1
3	-	edit2
1	+	# About me
2	+	
3	+	My name is Mona Lisa.

- Now that you have changes in a branch off of main, you can open a pull request.
- Pull requests are the heart of collaboration on GitHub. When you open a pull request, you're proposing your changes and requesting that someone review and pull in your contribution and merge them into their branch. Pull requests show differences of the content from both branches. The changes, additions, and subtractions are shown in different colors.
- As soon as you make a commit, you can open a pull request and start a discussion, even before the code is finished. In this step, you'll open a pull request in your own repository and then merge it yourself.

5. Open a pull request

- Click the Pull requests tab of your hello-world repository.
- Click New pull request.
- In the Comparisons box, select the branch you made, readme-edits, to compare with main (the original).
- Look over your changes in the diffs on the Compare page, make sure they're what you want to submit.
- Click Create pull request.
- Give your pull request a title and write a brief description of your changes. You can include emojis and drag and drop images and gifs.
- Click Create pull request.

6. Merge your pull request

- In this final step, you will merge your readme-edits branch into the main branch. After you merge your pull request, the changes on your readme-edits branch will be incorporated into main.
- Sometimes, a pull request may introduce changes to code that conflict with the existing code on main. If there are any conflicts, GitHub will alert you about the conflicting code and prevent merging until the conflicts are resolved. You can make a commit that resolves the conflicts or use comments in the pull request to discuss the conflicts with your team members.

6. Merge your pull request

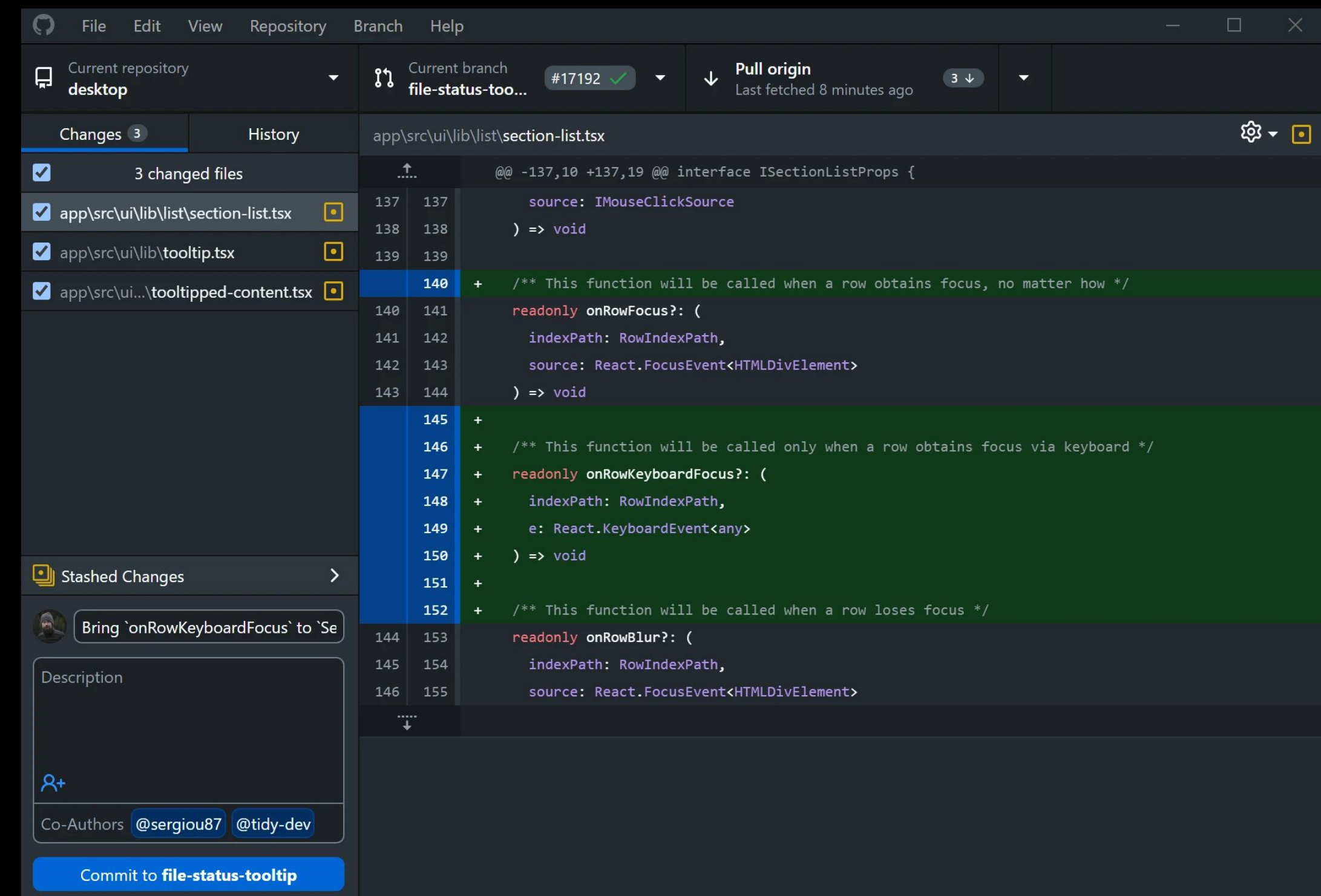
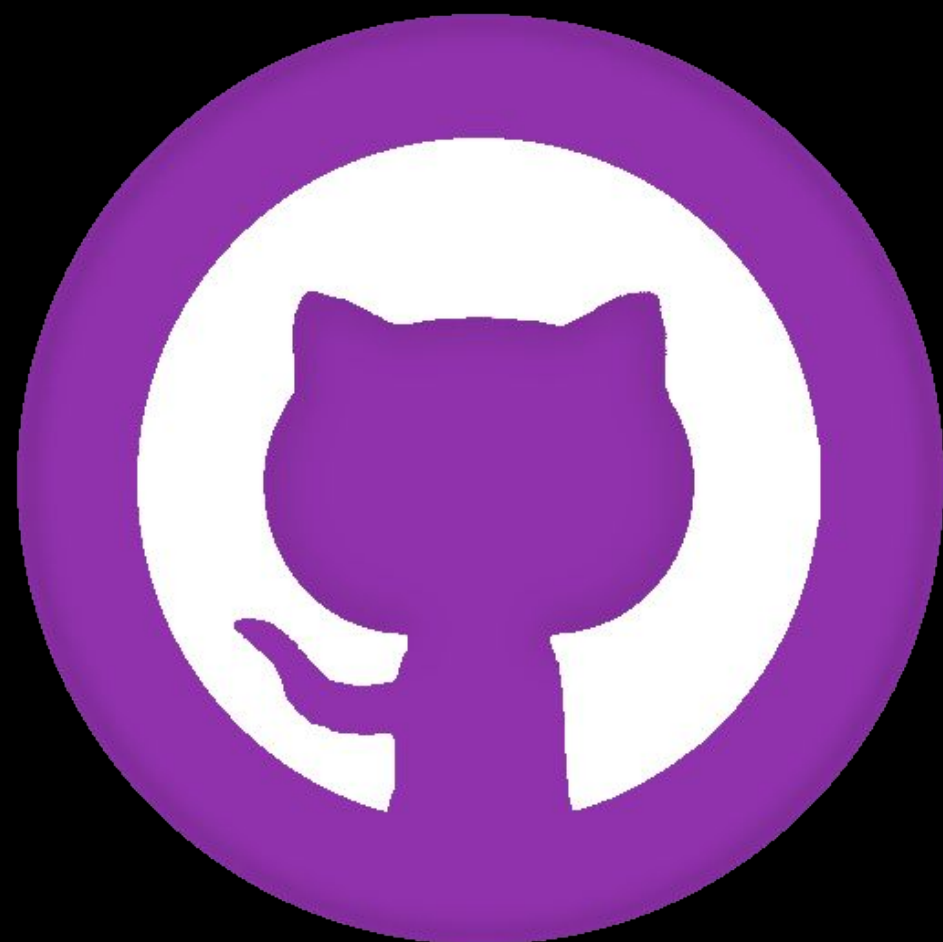
- At the bottom of the pull request, click **Merge pull request** to merge the changes into main.
- Click **Confirm merge**. You will receive a message that the request was successfully merged and the request was closed.
- Click **Delete branch**. Now that your pull request is merged and your changes are on main, you can safely delete the readme-edits branch. If you want to make more changes to your project, you can always create a new branch and repeat this process.
- Click back to the **Code** tab of your hello-world repository to see your published changes on main.

7. Setting up your environment

1. Download GitHub Desktop: <https://github.com/apps/desktop>

Instruction:

<https://docs.github.com/en/desktop/overview/getting-started-with-github-desktop>



7. Setting up your environment

2. Setting up Git locally

- Opening terminal (Cmd + Space and Type Terminal)
- Check if Git is already installed: `git --version`
 - Git for windows: <https://gitforwindows.org/>
 - If not, install using homebrew: `brew install git`
- Configure your Git
 - `git config --global user.name "Your Name"`
 - `git config --global user.email "your.email@example.com"`

7. Setting up your environment

2. Setting up Git locally

- Choose a Default Text Editor
 - Default: `git config --global core.editor "nano"`
 - `git config --global core.editor "code --wait"` (using VS code)
 - Review setting: `git config --list`
- Set up SSH for GitHub
 - (<https://docs.github.com/en/authentication/connecting-to-github-with-ssh>)
 - `ssh-keygen -t ed25519 -C "your.email@example.com"` or `ssh-keygen -t rsa -b 4096 -C "your.email@example.com"`
 - `cd ~/.ssh`; list the content by `ls -la` (should see `id_ed25519` and `id_ed25519.pub`)
 - `eval "$(ssh-agent -s)"`
 - `ssh-add ~/.ssh/id_ed25519` (Replace `id_ed25519` with `id_rsa` if you used RSA.)
 - Add the key on Github (`pbcopy < ~/.ssh/id_ed25519.pub`) (Settings > SSH and GPG keys -> New SSH key)
 - Test: `ssh -T git@github.com`

7. Setting up your environment

3. Git Clone

- git clone <https://github.com/username/repository.git> (Replace with actual URL)
- cd repository
- list the files with: ls

7. Setting up your environment

4. Git Init

- Create a New Project Folder: `mkdir my-new-project`
- Enter the New Project Folder: `cd my-new-project`
- Run: `git init` (create a hidden `.git` directory)
- Create a Python files: `print("Hello, Git!")`
- Stage the File: `git add hello.py`
- Commit the changes: `git commit -m "Initial commit: Add hello.py with a greeting function"`

7. Setting up your environment

5. Git Flow: Branching, Pulling, Pushing and Merging

- Create a branch: `git branch feature-update`
- Switch to the branch: `git checkout feature-update`
- Create and Switch: `git checkout -b feature-update`
- Edit the Python file
- Stage: `git add hello.py`
- `git commit -m "Update greeting message in hello.py"`
- Push the branch to the remote repository: `git push origin feature-update`
- Log in to Github; Review the changes; Create a pull request to merge feature-update into main
- Merge the Pull Request:
 - `git checkout main`
 - `git merge feature-update`
- Update the local main branch: `git pull origin main`

Exercise

Question: what is the role of Pull request and when you can skip Pull request?

Farmilize yourself with GitHub Flow:

<https://docs.github.com/en/get-started/using-github/github-flow>

Cheatsheet:

<https://training.github.com/downloads/github-git-cheat-sheet.pdf>

Reference Document: <https://docs.github.com/en>

Exercise

- Create a python file in your cloned git repository
- Create a markdown file that contains:
 - your expectation for this course
 - your general background
 - your knowledge about R
 - your knowledge about Python
 - try to use different syntax
- Push your folder to Github with source control in VS code

Next Step

- You can now clone a GitHub repository to create a local copy on your computer. From your local repository you can commit, and create a pull request to update the changes in the upstream repository. For more information, see "[Cloning a repository](#)" and "[Set up Git](#)."
- You can find interesting projects and repositories on GitHub and make changes to them by creating a fork of the repository. Forking a repository will allow you to make changes to another repository without affecting the original. For more information, see "[Fork a repository](#)."
- Each repository on GitHub is owned by a person or an organization. You can interact with the people, repositories, and organizations by connecting and following them on GitHub. For more information, see "[Be social](#)."
- GitHub has a great support community where you can ask for help and talk to people from around the world. Join the conversation on [GitHub Community](#).

Reading

Progenetix data:

<https://academic.oup.com/database/article/doi/10.1093/database/baab043/6323245>

Web:

<https://progenetix.org/>

Introduction to Markdown

README.md

Initialize this repository with:

☐ Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

- **GitHub combines a syntax for formatting text called GitHub Flavored Markdown with a few unique writing features.**
- **Markdown is an easy-to-read, easy-to-write syntax for formatting plain text**

Basic writing and formatting syntax

Headings

To create a heading, add one to six `#` symbols before your heading text. The number of `#` you use will determine the hierarchy level and typeface size of the heading.

```
# A first-level heading
## A second-level heading
### A third-level heading
```

A first-level heading

A second-level heading

A third-level heading

Basic writing and formatting syntax

Styling text [↗](#)

You can indicate emphasis with bold, italic, strikethrough, subscript, or superscript text in comment fields and `.md` files.

Style	Syntax	Keyboard shortcut	Example	Output
Bold	<code>** **</code> or <code>_ _</code>	<code>Command</code> + <code>B</code> (Mac) or <code>Ctrl</code> + <code>B</code> (Windows/Linux)	<code>**This is bold text**</code>	This is bold text
Italic	<code>* *</code> or <code>_ _</code>	<code>Command</code> + <code>I</code> (Mac) or <code>Ctrl</code> + <code>I</code> (Windows/Linux)	<code>_This text is italicized_</code>	<i>This text is italicized</i>
Strikethrough	<code>~~ ~~</code> or <code>~ ~</code>	None	<code>~~This was mistaken text~~</code>	This was mistaken text
Bold and nested italic	<code>** **</code> and <code>_ _</code>	None	<code>**This text is _extremely_ important**</code>	This text is <i>extremely</i> important
All bold and italic	<code>*** **</code>	None	<code>***All this text is important***</code>	<i>All this text is important</i>

Basic writing and formatting syntax

Quoting text [↗](#)

You can quote text with a `>`.

Text that is not a quote

`>` Text that is a quote

Quoted text is indented with a vertical line on the left and displayed using gray type.

Text that is not a quote

`|` Text that is a quote

Basic writing and formatting syntax

Quoting code [↗](#)

You can call out code or a command within a sentence with single backticks. The text within the backticks will not be formatted. You can also press the `Command + E` (Mac) or `Ctrl + E` (Windows/Linux) keyboard shortcut to insert the backticks for a code block within a line of Markdown.

```
Use `git status` to list all new or modified files that haven't yet been committed.
```

```
Use git status to list all new or modified files that haven't yet been committed.
```

To format code or text into its own distinct block, use triple backticks.

```
Some basic Git commands are:
```
git status
git add
git commit
```
```

```
```python```r
```



# Basic writing and formatting syntax

## Links

---

You can create an inline link by wrapping link text in brackets `[ ]`, and then wrapping the URL in parentheses `( )`. You can also use the keyboard shortcut `Command + K` to create a link. When you have text selected, you can paste a URL from your clipboard to automatically create a link from the selection.

You can also create a Markdown hyperlink by highlighting the text and using the keyboard shortcut `Command + V`. If you'd like to replace the text with the link, use the keyboard shortcut `Command + Shift + V`.

This site was built using `[GitHub Pages](https://pages.github.com/)`.



# Basic writing and formatting syntax

For more advanced writing and formatting syntax, please visit this website:

<https://docs.github.com/en/get-started/writing-on-github/working-with-advanced-formatting>

# Editing on Github Website

# Text formatting toolbar

Every comment field on GitHub contains a text formatting toolbar, allowing you to format your text without learning Markdown syntax. In addition to Markdown formatting like bold and italic styles and creating headers, links, and lists, the toolbar includes GitHub-specific features such as @mentions, task lists, and links to issues and pull requests.

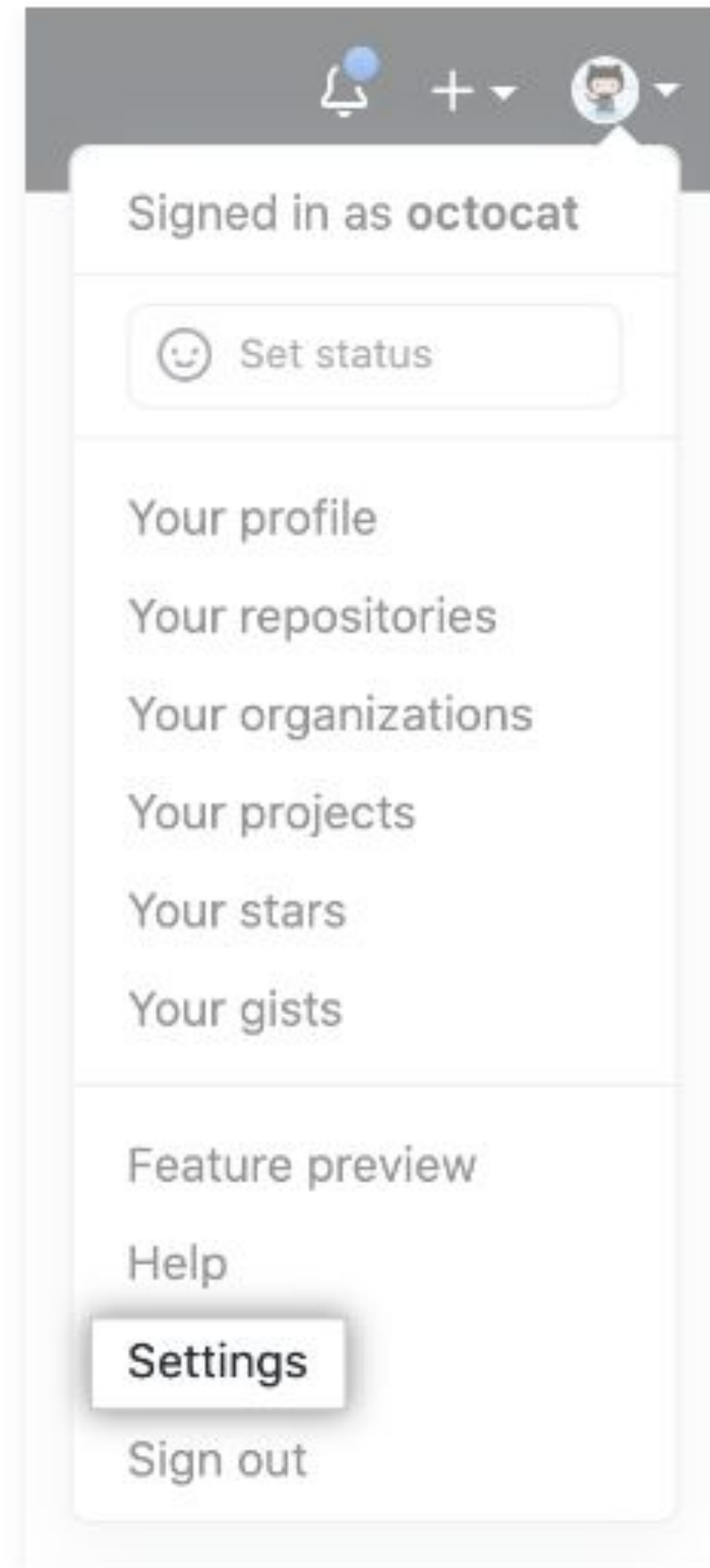
# Text formatting toolbar


You can enable a fixed-width font in every comment field on GitHub. Each character in a fixed-width, or monospace, font occupies the same horizontal space which can make it easier to edit advanced Markdown structures such as tables and code snippets.



# Text formatting toolbar

1 In the upper-right corner of any page, click your profile photo, then click **Settings**.



2 In the left sidebar, click  **Appearance**.



# Text formatting toolbar

## Enabling fixed-width fonts in the editor

- 3 Under "Markdown editor font preference", select **Use a fixed-width (monospace) font when editing Markdown**.

### Markdown editor font preference

Font preference for plain text editors that support Markdown styling (e.g. pull request and issue descriptions, comments.)

☐ **Use a fixed-width (monospace) font when editing Markdown**

# Visual Studio Code

# What is Visual Studio Code

- Free, open-source code editor developed by Microsoft
- Lightweight but powerful
- Supports many languages (Python, C++, JavaScript, R, etc.)
- Available on Windows, macOS, and Linux

# Installing VS Code

## 1. Download VS Code

Go to <https://code.visualstudio.com/> and Click “**Download for macOS**” (Intel or Apple Silicon depending on your chip)

## 2. Install



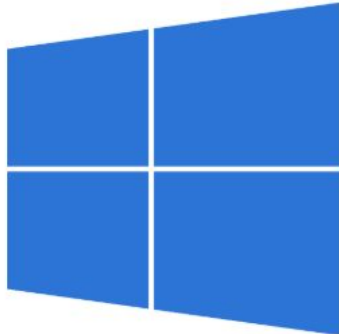
Drag “Visual Studio Code.app” to the Applications folder, making it available in the macOS Launchpad.

## 3. Open VS code

By double clicking the icon in the Applications folder

## Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.



↓ Windows

Windows 10, 11

User Installer [x64](#) [Arm64](#)

System Installer [x64](#) [Arm64](#)

.zip [x64](#) [Arm64](#)

CLI [x64](#) [Arm64](#)

↓ .deb

Debian, Ubuntu

.deb [x64](#) [Arm32](#) [Arm64](#)

.rpm [x64](#) [Arm32](#) [Arm64](#)

.tar.gz [x64](#) [Arm32](#) [Arm64](#)

Snap [Snap Store](#)

CLI [x64](#) [Arm32](#) [Arm64](#)

↓ .rpm

Red Hat, Fedora, SUSE

↓ Mac

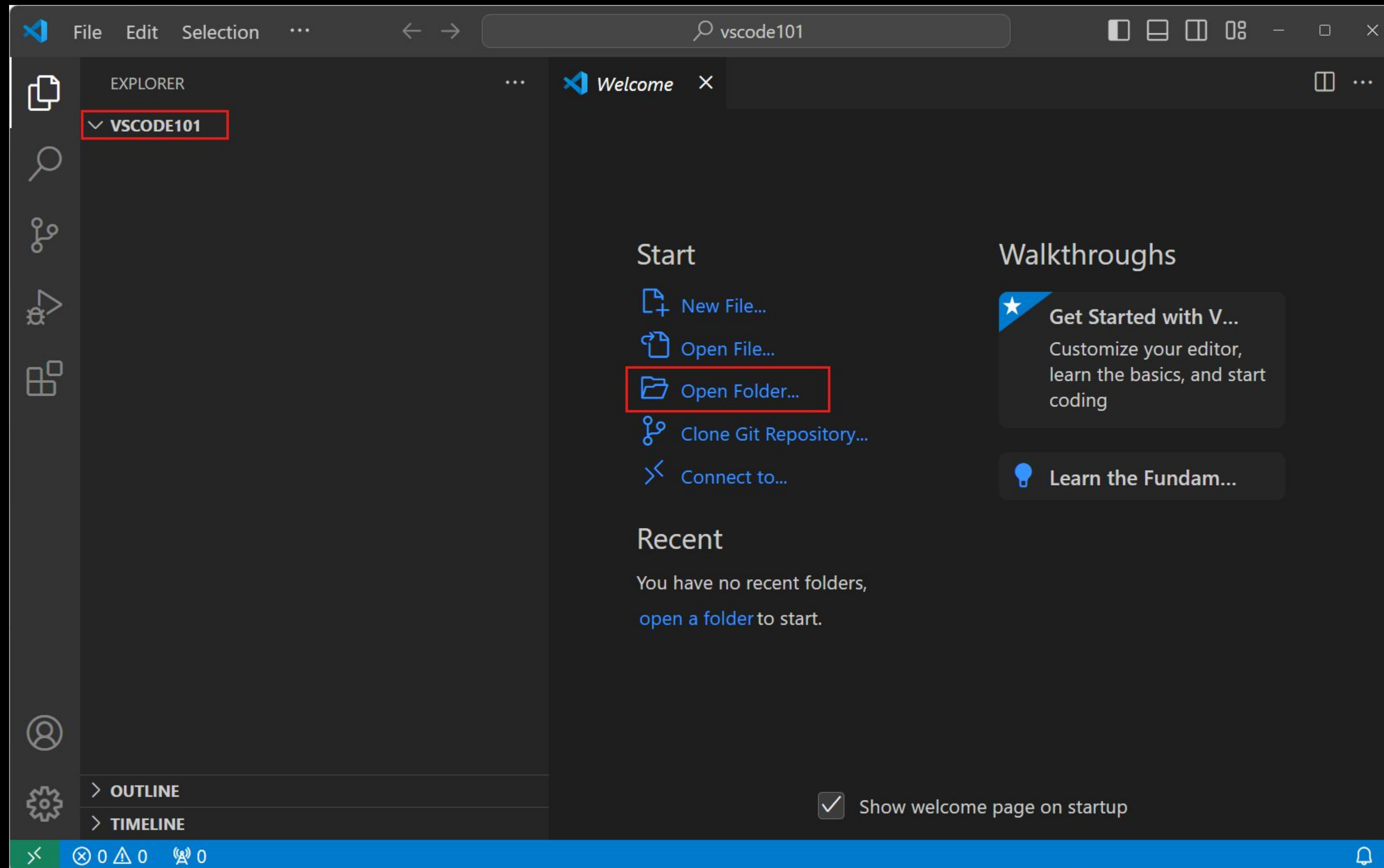
macOS 10.15+

.zip [Intel chip](#) [Apple silicon](#) [Universal](#)

CLI [Intel chip](#) [Apple silicon](#)

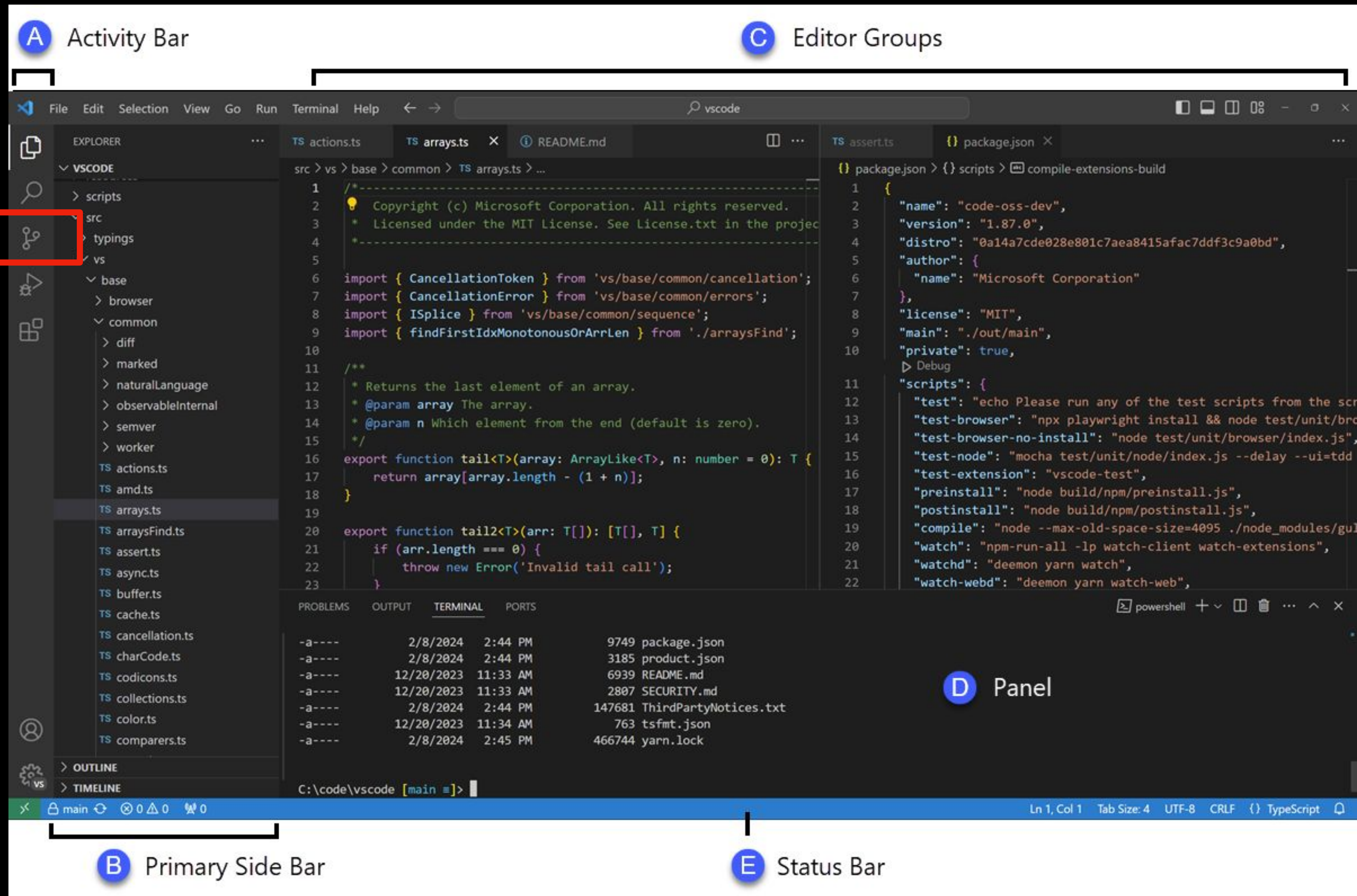
# Interface Overview

When you first open VS Code, you should see the Welcome page with different actions to get started.





# Interface Overview



# Setting Up Python in VS Code

1. Install Python in terminal
  - run the following line in your terminal: `brew install python`
2. Install the Python Extension in VS Code
  - Go to extensions (in the left bar)
  - Search for “Python” and install it
3. Create a folder and open a Python file
  - create a file with `.py` extension
  - VS code will prompt you to select the Python interpreter



# Connect GitHub to VS Code

- Go to the Source Control panel in the left sidebar
- Click “Sign in to GitHub”, it will open a browser to authorize your GitHub account. After you accept the permissions, it will link your GitHub with VS code
- Now you can open a repository from Github directly:  
press CMD + Shift + P, and then type “Git: Clone”  
Paste the Github repo URL that you just created in the Github website
- You can start to work in a Git repository, for more information about source control in VS code, please refer to <https://code.visualstudio.com/docs/sourcecontrol/overview>